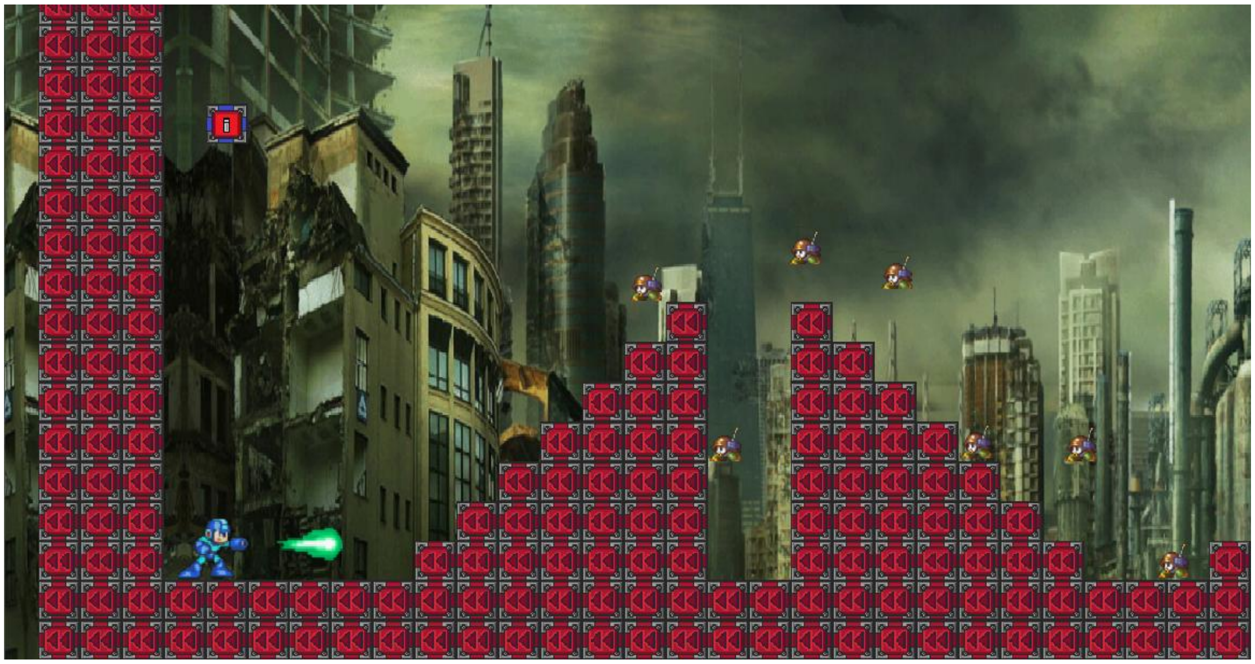
Mega Man DirectX Manual



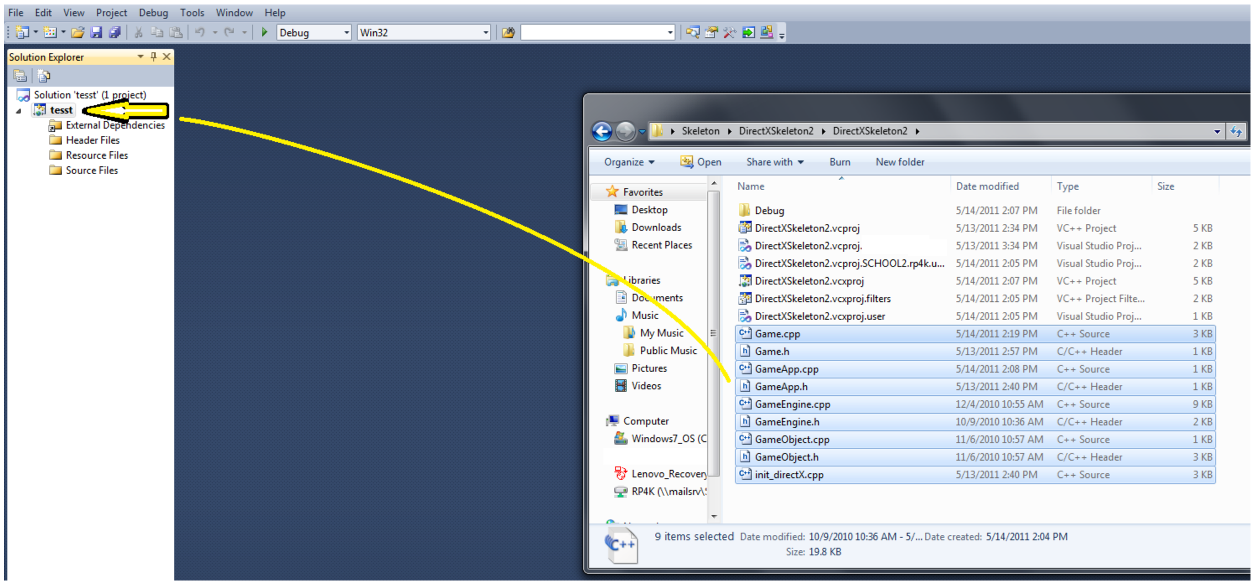
The first thing I would suggest to do with the student is to make a console application, and teach them the basic syntax of C++. Since Mega Man is the introductory C++ class, expect your students to know nothing about pointers or header files.

What I like to do is show why pointers are fun, yet very dangerous, by having a while(true) {new ..} loop that allocates memory until your computer crashes. When you open up task manager, you can see your memory usage skyrocketing. This shows the importance of managing your memory, and to stress that all pointers must be deleted somewhere.

Once the student has a decent understand of C++ syntax, get them to create their project for their game. **(DIRECTX MUST BE INSTALLED FOR ANY OF THIS TO WORK)**. This should also be programmed on Microsoft Visual C++ 2010.

1.0 Preparing the Skeleton

1. Make an empty C++ project. Call it MegaMan. WARNING: Making the project in any drive other than C:/ tends to break Visual Studio
2. Drag and drop the .cpp and .h files from the provided skeleton into the solution explorer. It should be dropped on the name of the project. You know you have put them in the right spot if the header and solution files are sorted into different folders.

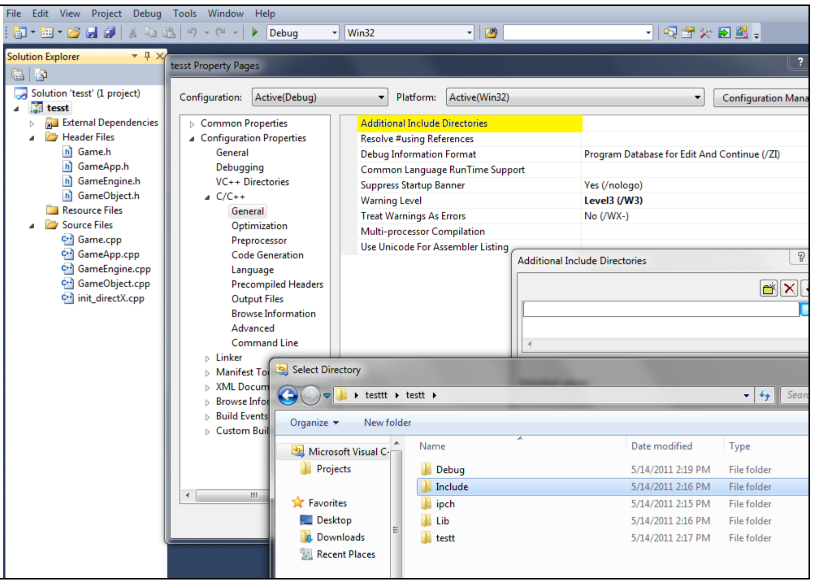


Name	Date modified	Type	Size
Debug	5/14/2011 2:07 PM	File folder	
DirectXSkeleton2.vcproj	5/13/2011 2:34 PM	VC++ Project	5 KB
DirectXSkeleton2.vcproj.	5/13/2011 3:34 PM	Visual Studio Proj...	2 KB
DirectXSkeleton2.vcproj.SCHOOL2.rp4k.u...	5/14/2011 2:05 PM	Visual Studio Proj...	2 KB
DirectXSkeleton2.vcxproj	5/14/2011 2:07 PM	VC++ Project	5 KB
DirectXSkeleton2.vcxproj.filters	5/14/2011 2:05 PM	VC++ Project Filte...	2 KB
DirectXSkeleton2.vcxproj.user	5/14/2011 2:05 PM	Visual Studio Proj...	1 KB
Game.cpp	5/14/2011 2:19 PM	C++ Source	3 KB
Game.h	5/13/2011 2:57 PM	C/C++ Header	1 KB
GameApp.cpp	5/14/2011 2:08 PM	C++ Source	1 KB
GameApp.h	5/13/2011 2:40 PM	C/C++ Header	1 KB
GameEngine.cpp	12/4/2010 10:55 AM	C++ Source	9 KB
GameEngine.h	10/9/2010 10:36 AM	C/C++ Header	2 KB
GameObject.cpp	11/6/2010 10:57 AM	C++ Source	1 KB
GameObject.h	11/6/2010 10:57 AM	C/C++ Header	3 KB
init_directX.cpp	5/13/2011 2:40 PM	C++ Source	3 KB

9 items selected Date modified: 10/9/2010 10:36 AM - 5/... Date created: 5/14/2011 2:04 PM
Size: 19.8 KB

3. Change your projects includes, libraries, and dependencies

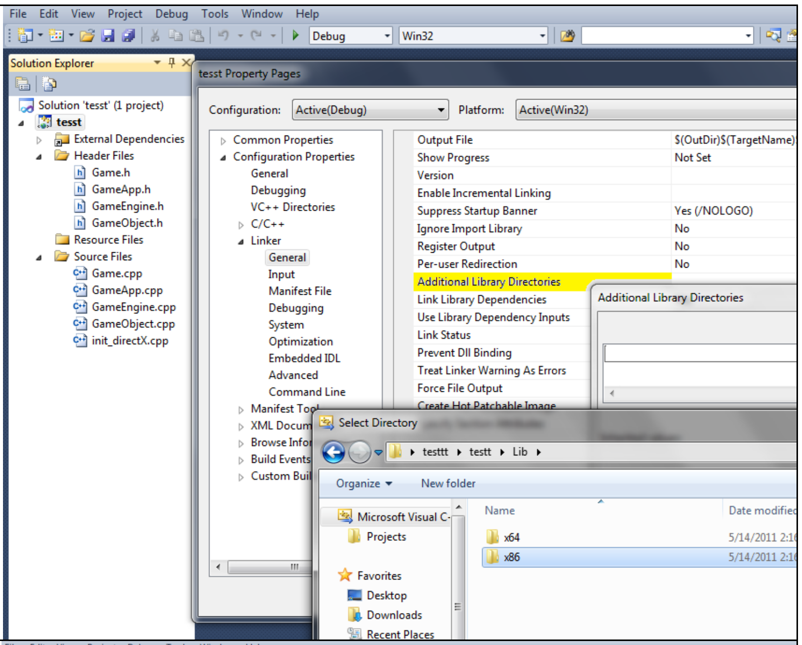
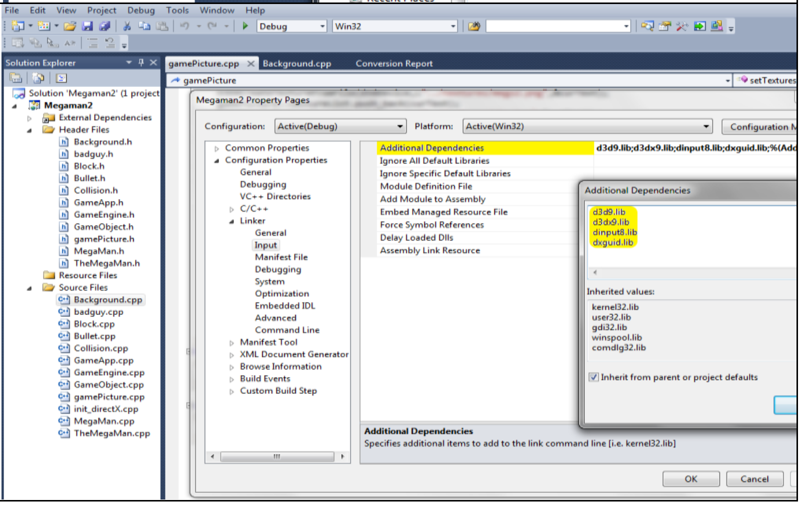
a. **Includes**: project -> properties -> configuration properties -> C/C++ -> general -> Additional Include Directories. The file path that should go here is "C:/…/skeleton/DirectXSkeleton2/include"



Configuration: Active(Debug)
Platform: Active(Win32)

Additional Include Directories	
Resolve #using References	
Debug Information Format	Program Database for Edit And Continue (/ZI)
Common Language RunTime Support	
Suppress Startup Banner	Yes (/nologo)
Warning Level	**Level3 (/W3)**
Treat Warnings As Errors	No (/WX-)
Multi-processor Compilation	
Use Unicode For Assembler Listing	

Name	Date modified	Type
Debug	5/14/2011 2:19 PM	File folder
Include	5/14/2011 2:16 PM	File folder
ipch	5/14/2011 2:15 PM	File folder
Lib	5/14/2011 2:16 PM	File folder
testt	5/14/2011 2:17 PM	File folder

b. **Libraries: project -> properties -> configuration properties ->Linker-> general -> Additional Include Directories.** The file path that should go here is “C:/…/skeleton/DirectXSkeleton2/lib/x86”	
c. **Dependencies**: **project -> properties -> configuration properties ->Linker-> input -> Additional Dependencies.** The information here should be: “d3d9.lib; d3dx9.lib; dinput8.lib; dxguid.lib;”	

tesst Property Pages

Configuration: Active(Debug) Platform: Active(Win32)

Output File	$(OutDir)$(TargetName)
Show Progress	Not Set
Version	
Enable Incremental Linking	
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	
Link Library Dependencies	
Use Library Dependency Inputs	
Link Status	
Prevent Dll Binding	
Treat Linker Warning As Errors	
Force File Output	
Create Hot Patchable Image	

Select Directory: testtt ▸ testt ▸ Lib ▸

Name	Date modified
x64	5/14/2011 2:1
x86	5/14/2011 2:1

Megaman2 Property Pages

Configuration: Active(Debug) Platform: Active(Win32)

Additional Dependencies	d3d9.lib;d3dx9.lib;dinput8.lib;dxguid.lib;%(Add
Ignore All Default Libraries	
Ignore Specific Default Libraries	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded Dlls	
Assembly Link Resource	

Additional Dependencies:

d3d9.lib
d3dx9.lib
dinput8.lib
dxguid.lib

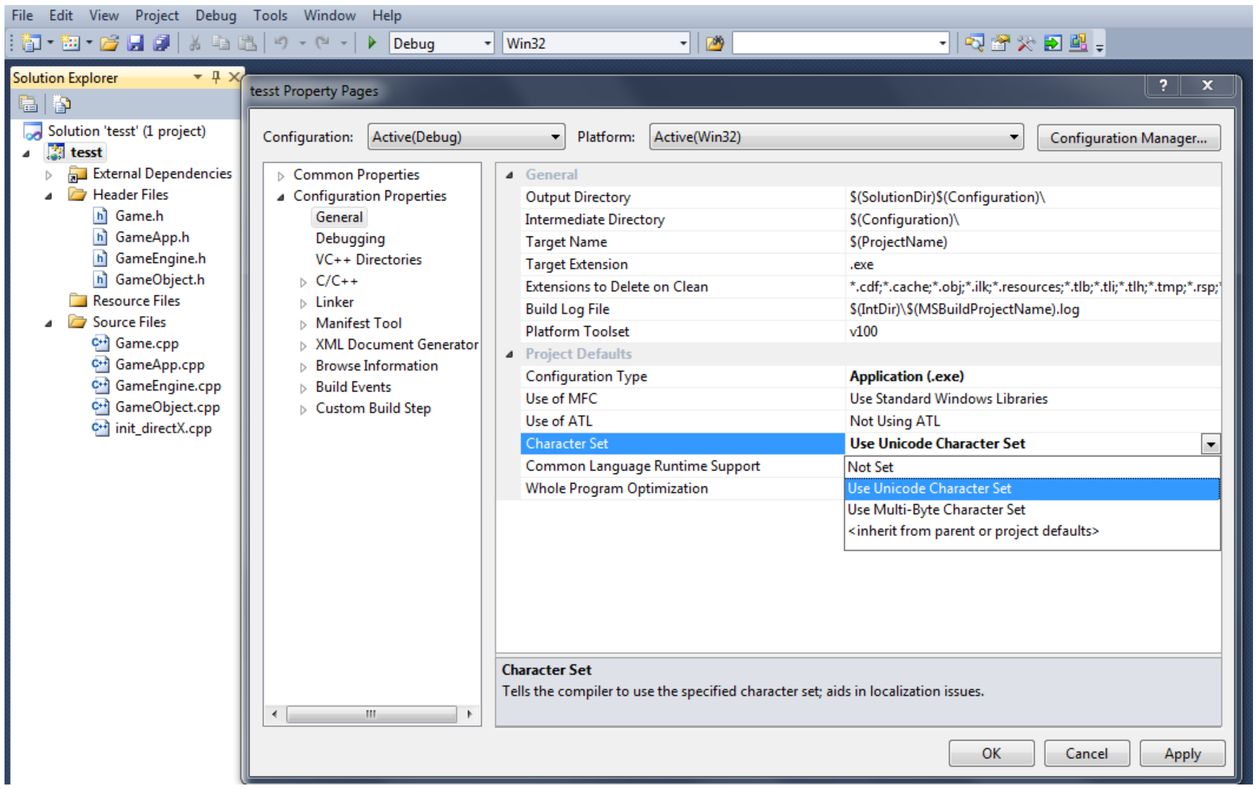
Inherited values:

kernel32.lib
user32.lib
gdi32.lib
winspool.lib
comdlg32.lib

☑ Inherit from parent or project defaults

Additional Dependencies
Specifies additional items to add to the link command line [i.e. kernel32.lib]

4. Change the **character set** of your project (should be Unicode)
project -> properties -> configuration properties -> general -> Character Set.



Configuration: Active(Debug) Platform: Active(Win32)

General	
Output Directory	$(SolutionDir)$(Configuration)\
Intermediate Directory	$(Configuration)\
Target Name	$(ProjectName)
Target Extension	.exe
Extensions to Delete on Clean	*.cdf;*.cache;*.obj;*.ilk;*.resources;*.tlb;*.tli;*.tlh;*.tmp;*.rsp;
Build Log File	$(IntDir)\$(MSBuildProjectName).log
Platform Toolset	v100
Project Defaults	
Configuration Type	**Application (.exe)**
Use of MFC	Use Standard Windows Libraries
Use of ATL	Not Using ATL
Character Set	**Use Unicode Character Set**
Common Language Runtime Support	
Whole Program Optimization	

Not Set
Use Unicode Character Set
Use Multi-Byte Character Set
<inherit from parent or project defaults>

Character Set
Tells the compiler to use the specified character set; aids in localization issues.

5. Press play! Everything should compile, link and run at this point. A black box should appear on your screen.

BEFORE YOU TEACH THIS CLASS!
Get to know this skeleton very well, because there is a lot of code in there. Most of it is from the internet, so the coding style may seem foreign. (there are some gotos).
Once you fully understand how all the classes work, and all the methods and variables inside of them, then you can start putting objects on the screen!

2.0 Creating Objects

Objects in Mega Man are created using the GameObject class. As the skeleton stands, the GameObject class has the ability to create and render a white square, but it is missing several parts of its functionality. Later on, the GameObject class will be beefed up to be able to rotate images, scale them dynamically, and render textures on top of them.

For this lesson, all we want to do is get an object on the screen. All of this work has to be done inside the Game class. Get the student to define a new GameObject variable in the Game's header file.

```
.
.
.
        int createGameObjects();

private:
        char keysState[256];                // the state of the 256 keyboard keys
        DIMOUSESTATE mouseState;            // the state of the mouse

        // procsesses the keyboard input from the user
        int processKeyboardInput(void);
        int processMouseInput(void);
        GameObject* megaMan;
};
```

This is a good point to reiterate that this game object is a pointer, and must be deleted in the Game's destructor. Also, take the student through the Game class now, and show them all of the steps of the program. Start with initialization, rendering, updating, and end with destruction. So the next step would be to initialize obj1. Take the student to createGameObjects().

```
int Game::createGameObjects()
{
        megaMan = new
        GameObject(D3DXVECTOR2(100,100),D3DXVECTOR2(100,100),gameEngine.d3dDev);

        return 0;
}
```

Finally, all we have to do is render the object, and delete it in the destructor. This is also done in the Game class.

```
int Game::RenderFrame(void)
{
	// clear the window to a deep blue
	gameEngine.d3dDev->Clear(0, NULL, D3DCLEAR_TARGET, D3DCOLOR_XRGB(0, 0, 0), 1.0f,
0);

    gameEngine.d3dDev->BeginScene();    // begins the 2D scene

	// TODO: Render Code Here

	megaMan->render(gameEngine.d3dDev);

	gameEngine.d3dDev->EndScene();    // ends the 2D scene

    gameEngine.d3dDev->Present(NULL, NULL, NULL, NULL);   // displays the created frame
on the screen
	return 0;
}

Game::~Game(void)
{
	delete megaMan;
}
```

Run the game and you should get this!